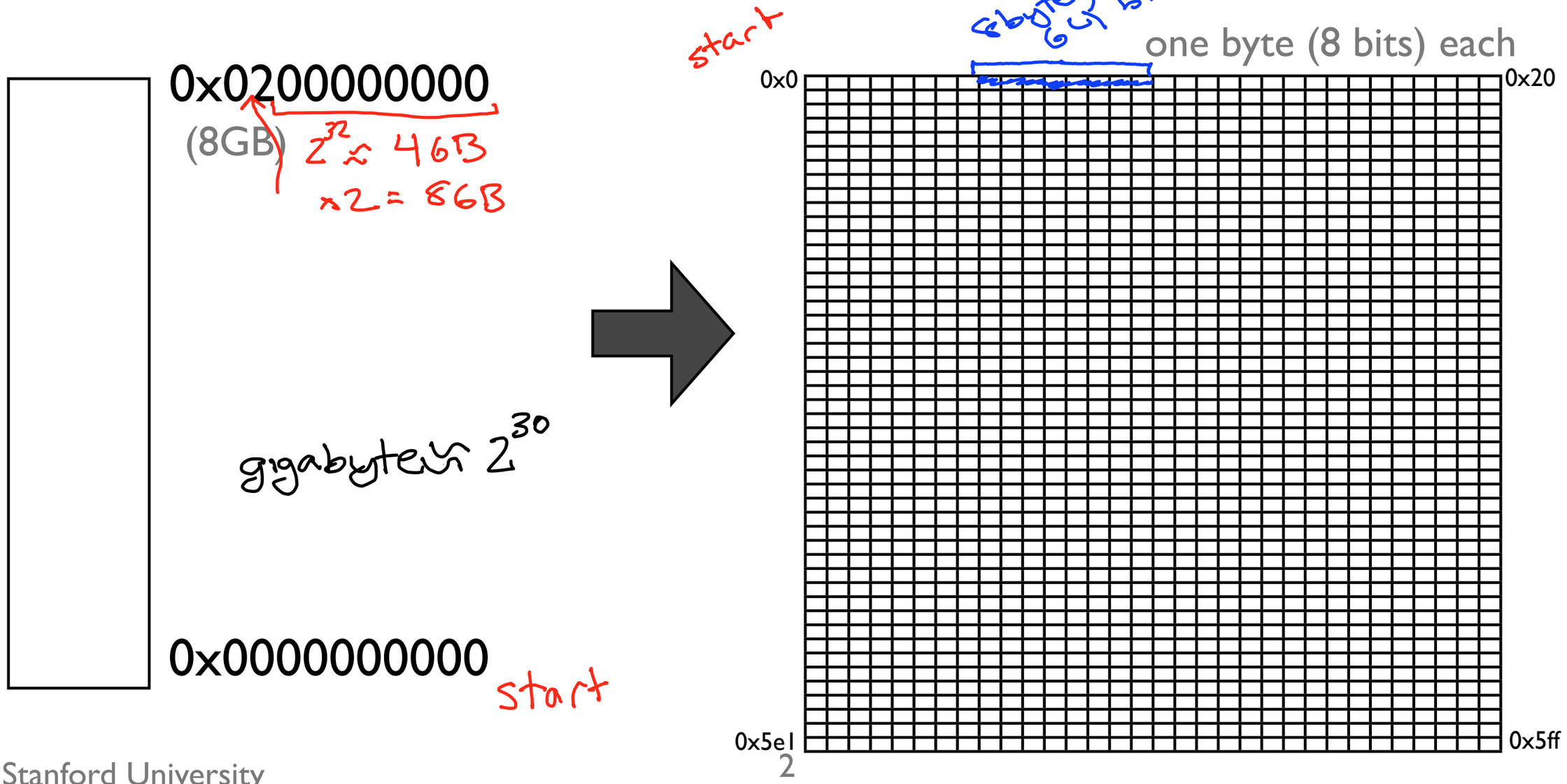
Memory, Byte Order, and Packet Formats

Computer Memory



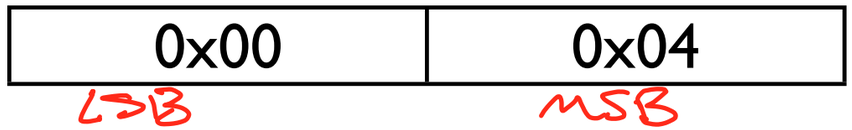
Endianness

1,024 = 0x0400 =

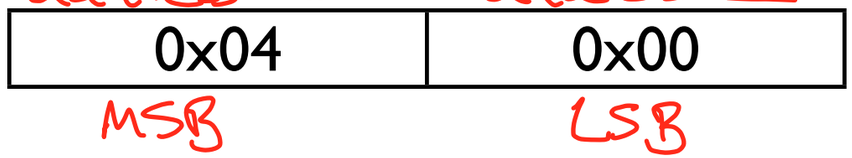
- Multibyte words: how do you arrange the bytes?
- Little endian: least significant byte is at lowest address
  - Makes most sense from an addressing/computational standpoint

LSB = least significant byte

MSB = most significant byte



- Big endian: most significant byte is at lowest address
  - Makes most sense to a human reader



Quiz

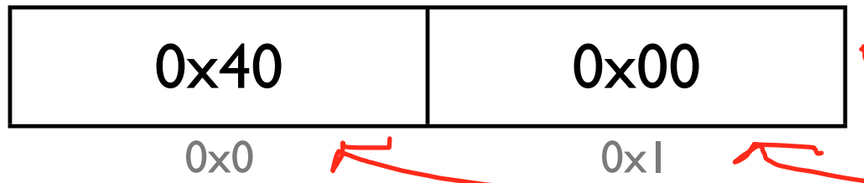
For each number, mark whether the hexidecimal representation is big endian or little endian. Don't use a calculator or other tool!

Width	Decimal	Hexidecimal	Big Endian	Little Endian
16 bits	53	0x3500		✓
16 bits	4116	0x1014	✓	
32 bits 5×2^{24}	5	0x00000005	✓	
32 bits ↳	83,886,080	0x00000005		✓
32 bits	305,414,945	0x21433412		✓

Network Byte Order

- Different processors have different endianness
  - Little endian: x86, big endian: ARM
- To interoperate, they need to agree how to represent multi-byte fields
- Network byte order is big endian

1,024 = 0x400 =



network order for 1,024

```
uint16_t val = 0x400;
uint8_t* ptr = (uint8_t*)&val;
```

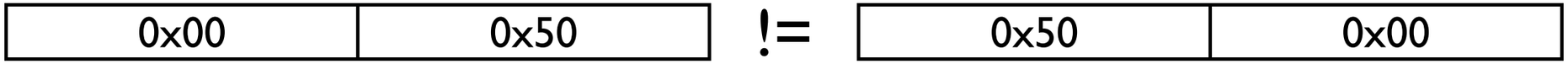
simple C code to test for endianess of processor

```
if (ptr[0] == 0x40) {
  printf("big endian\n")
}
else if (ptr[1] == 0x40) {
  printf("little endian\n");
}
else {
  printf("unknown endianness!\n");
}
```

Portable Code

- You have to convert network byte order values to your host order
- E.g., packet has a 16-bit port in network byte order, you're using a struct to access it, you want to check on your x86 processor if the port is 80

```
uint16_t http_port = 80; // Host order
if (packet->port == http_port) { ... // Network vs. host order
```



- Helper functions: htons(), ntohs(), htonl(), ntohl()
  - htons: "host to network short", ntohl: "network to host long"
  - #include <arpa/inet.h>

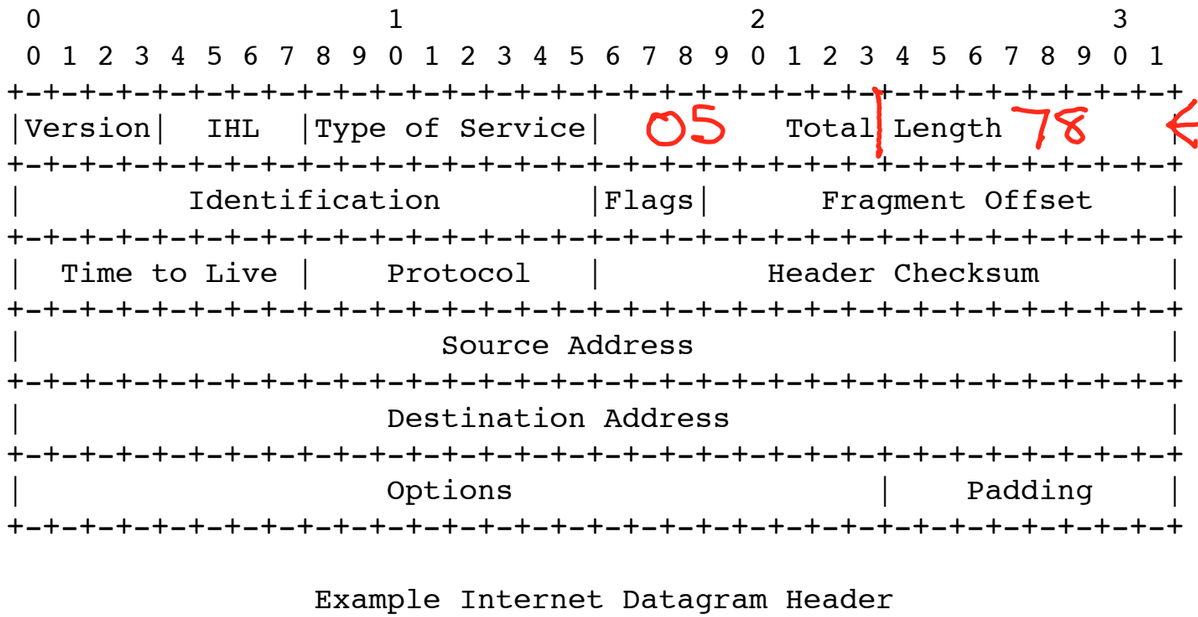
```
uint16_t http_port = 80; // Host order
uint16_t packet_port = ntohs(packet->port); ← swaps bytes!
if (packet_port == http_port) { ... // OK
```

Be careful whenever you handle network data!

Otherwise you will waste many (avoidable) hours debugging your code due to forgetting to convert or converting twice.

Packet Formats

maximum length: 65,535 bytes
1400 byte packet: 0x0578



Example Internet Datagram Header

RFC 791
(standard IETF illustration)

V	IHL	TOS	destination port
Identification		flags	fragment offset
TTL	Protocol	Header Checksum	
Source Address			
Destination Address			

32 bits (4 octets)

course note equivalent